Temporally-threaded Workspace: A Model for Providing Activity-based Perspectives on Document Spaces

Bor3 5290

Koichi Hayashi, Takahiko Nomura, Tan Hazama,
Makoto Takeoka, Sunao Hashimoto
Corporate Research Labs., Fuji Xerox Co., Ltd.
430 Sakai, Nakai-machi, Ashigarakami-gun,
Kanagawa 259-01, Japan
E-mail: hayashi.koichi@fujixerox.co.jp

Stephan Gudmundson
The Department of Computer Science
University of British Columbia
201-2366 Main Mall
Vancouver, B.C., V6T 1Z4, Canada

PD. 20. 06. 98

XP-000868462

p.87—96 = (10)

ABSTRACT

In this paper, we present a framework for providing *activity-based perspectives* of a document space, especially in the WWW. An activity-based perspective is a view of the subspace of the WWW document space that a knowledge worker should understand or modify while executing the activity. We designed the framework to reduce the cognitive overhead of managing document spaces dependent on various internal and external changes. Changes within the activity (often resulting from the natural progress of the activity) result in changes of focus in the subspace related to the activity. For such internal changes, we introduce a *temporally-threaded workspace* model. Our model introduces a structured workspace that maintains a thread of snapshots of a knowledge worker's perspective on a document space. Such threads of snapshots are constructed by monitoring user actions. External changes (for example, changes to documents managed in external sites) are independent of the progress of users' activities. To deal with these changes, we introduce a proxy mechanism to maintain documents in the same state as accessed. This paper also describes the implementation of prototype systems, in the WWW environment, based on our frameworks. *Interlocus* is a client/server system providing facilities based on the temporally-threaded workspace model. It provides a user interface that presents spatial-temporal views of a workspace thread. *Packrat* is a WWW proxy server that maintains documents in the same state as accessed.

KEYWORDS: hypertext, WWW, spatial hypertext, version management, shared workspace, authoring, activity

INTRODUCTION

Recent widespread adoption of the WWW makes hypertext authoring everyday work for many office workers. This means that common workers can prepare HTML documents and link to other documents with a standard word processor. Further, it means that the actions of workers handling local or shared documents affect the hypertext of the WWW-based Intranet in their organization. Specifically, by adding bookmarks to their favorite URL collections or by sending mail containing URLs to colleagues they are implicitly organizing a tiny part of the Intranet. From this perspective, office workers' everyday activities are viewed as continuous hypertext authoring processes.

Although this characterization enlarges the range of authoring, the essential characteristic of authoring has not changed. Marshall and Shipman [19] describe authoring in the following excerpt: "Authors collect and structure materials to reflect their own understanding or in anticipation of reader's possible interests, needs, or ability to comprehend the substrate of interrelated content." Thus, it is essential for knowledge workers to contribute their own perspectives on the retrieved documents. In today's WWW-centric world, every activity that involves retrieving, accessing, editing, or distributing documents must also involve authoring, by our characterization. Therefore, it is essential for workers to manage an editable local subspace of the WWW related to their activities.

With respect to authoring hypertext, Conklin [6] identified the problem of cognitive overhead as "the additional effort and concentration necessary to maintain several tasks or trails at one time." Spatial hypertext, investigated with VIKI [18], enables users to see the structure of hypertext via a two-dimensional representation. However, in these discussions, the graphical representation of the hypertext is concerned only with the current state. For the problem of maintaining a dynamic document subspace (i.e. the space associated with an activity), we need to consider another dimension: time.

When performing intellectual activities, internal and external changes make the activity-centric perspective on a document space more difficult to maintain. For example, suppose one must write a report about international CO_2 emissions control related to the company's products. A technical staff member would start by searching for related articles and papers from public sites, and collecting internal data about the company's products. He or she

HyperText 98 Pittsburgh PA USA
Copyright ACM 1998 0-89791-972-6/98/ 6...$5.00

would then analyze the resulting documents and identify the important ones. From these, he or she can draw conclusions, and organize the final report by adding the result of analysis and opinions. This process continuously affects a structure that is part of the WWW. Figure 1 illustrates internal and external changes of a document space.

Changes within the activity, resulting from the natural progress of the activity, often change the focus of the activity's space. In this example, while gathering information, the set of retrieved documents is the focus of the activity. While organizing the final report, selected documents and the user's own documents are the focus of the activity. On the other hand, the process of authoring does not always proceed in a predetermined way, as pointed by Streiz [21]. For example, when composing the final report, additional searching might be needed to get data that agrees with the logic of the report. Therefore, workers must handle changes to the focus of the space at different stages of the activity, while still understanding the whole structure, as it relates to the whole process.

External changes, such as changes to documents in external sites, are independent of the progress of workers' activities. The results of an activity often depend on external circumstances. In the above example, the conclusion of the report depends on the latest information about international agreement on CO_2 emissions controls. Documents from sites publishing such information might be altered regularly, and virtual documents generated dynamically could be different every time. If the contents of external documents change, the document space that the worker used becomes different from the current state. The document space that lead to the conclusion becomes obscure as time passes. To preserve the correct document space, the worker must store external documents in some sort of manageable storage.

In this paper we present a framework to help users maintain their *activity-based perspective* of document spaces. These document spaces include external documents, which are in the WWW of the Internet, and internal documents, which are stored in the WWW-based Intranet and the local storage. Our framework provides spatial-temporal views to reduce the cognitive overhead of managing document spaces sensitive to internal and external changes of activity related information.

In the following sections, first we review related work about perspectives of information spaces for hypertext authoring. We then introduce our framework, which includes the *temporally-threaded workspace* model for dealing with internal changes, and a proxy mechanism for dealing with the external changes. Further, we describe Interlocus and Packrat, experimental implementations of our framework. We then discuss our framework and the implementations, and compare our approach to the related work. Finally, we offer future work and conclusions.

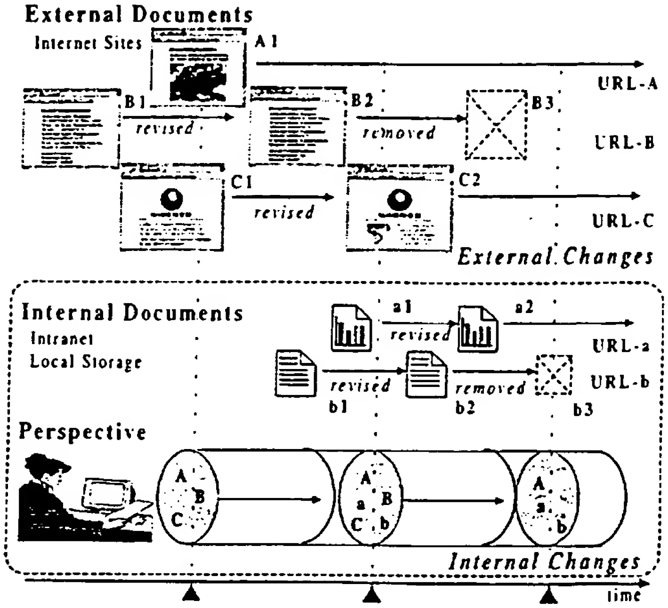

Figure 1: Internal changes and external changes

RELATED WORK

We provide a framework to help users obtain a spatial-temporal perspective on the document space related to their activities. The main features of our framework are graphical workspace in which workers organize documents related to their tasks, and a tool for managing the states of these dynamic workspaces and the documents themselves.

Spatial perspective of information space

For organizing pieces of information desk-like interfaces, onto which users can place small objects representing information, are among the most familiar graphical user interfaces. This way of using two-dimensional space is called desktop metaphor. The desktop metaphor adopted for major windowing system was influenced by Malone's cognitive analysis of using piles of paper on a desk [17]. A desktop maintains necessary documents for executing users' tasks and reminds the users of the state of the tasks. Rooms[4][15] further provides special desktops for each different type of task. Since some room contains the common documents and tools, a user can start on the same type of job without special preparation.

The concept of a workspace for managing personal information has evolved into shared workspaces that support collaborative work in shared document space. BSCW[1][2] provides a shared workspace in the WWW. A BSCW workspace maintains a set of documents and records any actions performed against documents in the space. It does not provide the desktop interface. Desktop interfaces that can be used on the WWW are rushing to the market and are called NUI (network user interface) [12]. NUI enables a more seamless world that does not distinguish between accessing local storage and accessing sites on the WWW.

For presenting the worker's view of the information space, a desktop provides limited sorts of clues, i.e. image, name, and position of icons. The spatial hypertext concept, explored with VIKI [18][19][20], presents another way of using two-dimensional space for placing pieces of information. VIKI provides richer visual components than a conventional desktop or hypertext graph browser. They can be clues for understanding and presenting one's own viewpoint within the two-dimensional space of the collected information.

As present commercial products have shown, the use of these spatial views is effective for reducing the cognitive overhead of understanding a document space; however, these systems are only concerned with the current state. We extend the model to handle temporal changes of the space.

Activity-based perspective of information space

Several studies on hypertext authoring present facilities for maintaining the worker's perspective on what they are doing during the process of authoring. The authoring model from Sepia[21] involves altering a shared hypertext space from separated views (activity spaces). It includes the content space, rhetorical space, planning space, and argumentation space. Users author hypertext by traversing these spaces. COWFISH [24] is a descendant of Sepia that enables users to define custom activity spaces. With these activity spaces, users can maintain the current view related to each activity.

Nelumbo[14]'s surface browser presents perspective of the progress of authoring. Nelumbo is a hypertext-based authoring tool whose authoring model generates a conventional tree document structure by composing tree parts extracted from the "underground" network hypertext structure. These tree parts are extracted explicitly by user's direct editing operations, and implicitly by user's history of accessing hypertext structure. By placing portions of the tree on the surface plane, the surface browser presents the state of authoring.

Techniques for managing changes to complex structures are the focus of research of version management for several fields, such as OODB[5][16] and hypertext[3][11]. Østerbye[23] discusses general techniques for version management for hypertext. Further, CoVer[7][8][9] and VerSE [10] introduce tasks for representing internal views of version management. These systems enable users to access the snapshot of the hypertext related to the state of a task at a certain date.

These systems are designed to support closed hypertext spaces and predetermined types of activities. We need additional facilities for supporting the enlarged authoring processes in today's WWW-centric world.

TEMPORALLY-THREADED WORKSPACE

In this section we describe our model for representing the document space related to a given activity. In nature these subspaces are not static; they continuously change according to the progress of the activity. Further, many of these changes are not predictable. The solution for dealing with these ever-changing spaces should satisfy following requirements:

- It must handle changes to the document space related with flexible processes of activities. Since many intellectual tasks proceed in flexible way and dynamically choose the activity's path during execution.
- Users can build perspectives of their activities without extra preparation. The user should not be bothered with specifying or declaring their activities other than doing actions inherent to their activities, such as gathering information and writing documents.
- Users can manage document spaces related with both current focus and whole process of an activity. The perspective of the current-focus space is useful for managing the progress of an activity; past document spaces are helpful for unpredictable events like backtracking.

In this paper we use the word *activity* to mean the human process of changing a hypertext space to achieve some specific goal. For example, preparing a report about international CO_2 emissions control, discussing the design of an environmentally safer product, and planning customer support using the Internet are all activities. We do not deal with predetermined categories of activity, such as writing reports, discussing technical issues, and planning projects, since we intend to cover a wide range of knowledge workers' activities, including ad-hoc tasks.

We introduce a model of *temporally-threaded workspaces* to capture the changes to the document space that are related to an activity. An *activity-based perspective* of a document space is a view of a group of interrelated workspace threads. A workspace thread is designed as an environment for providing the necessary set of documents and tools for carrying out an activity. As Malone's work and Marshall's work suggest, the set of gathered documents placed in a spatial workspace reflects the worker's understanding of an activity at a given state. We use this finding to enable freezing document spaces without workers' extra efforts.

To capture the activity as a whole and each state of the activity, we record a sequence of changes to the workspace as a thread. The latest snapshot of workspace in a thread represents the present focus of the activity-related document space. A workspace thread enables workers to restart with the past stage of the activity, to reuse documents from their own or others' past activities, and to see the progress of their own or others' activity. Note that this model only manages organizations of references to documents. For managing document contents and versions we introduce another mechanism in the next section.

Objects

In this section, we present a more formal description of the objects used in constructing an activity-based perspective. Figure 2 illustrates relationships among these objects. An activity-based perspective is defined as a pair of a set of (workspace) *threads* and a set of *links*.

Perspective = (threads, links)

A thread represents a workspace thread, which is a sequence of changes to the set of documents necessary for performing an activity. A thread is defined as a set of snaps bound with creation time. Figure 2 shows three threads.

Thread = (creation-time, snap)*

A snap (snapshot) represents a focused area in the subspace related to an activity. A snap holds the current state of the set of documents shown in the workspace or a recorded state from a specific time in the past. A snap holds the position of each document placed in the workspace. A snap is defined as set of pairs of anchor and the position of the anchor. The first thread in Figure 2 has three snaps.

Snap = (anchor, position)*

An anchor is a unit to represent an activity-dependent value added to an external document. An external document may be a node in the hypertext (e.g. the WWW) in which the workers perform their activities. Anchor is defined as a triple of name, image, and locator. A name is a string describing the meaning of a document under the workspace. An image is a graphical spatial element that expresses the characteristics of the referenced document like an icon in desktop metaphor. A locator is a reference to an external document. URL is an example of a locator. The first snap of the first thread in Figure 2 has four anchors.

Anchor = (name, image, locator)

Like nodes in conventional hypertext, workspace threads may be linked to each other. As links among workspace threads, we introduce directed one-to-one links. A link is defined as a pair of workspaces, the source and the destination.

Link = (source-thread, destination-thread)

We provides two types of links: reference link and continuation link. Both links are prompted by explicit or implicit user operations. A reference link describes the relation that a certain activity is related to another activity. In other words, while performing a certain activity, a worker needed to reference the other activity. A continuation link records the fact that the activity is a continuation from another activity. In other words, while performing some activity, a worker started subtask by creating successive new workspace thread. The structure formed by the continuation links becomes a tree.

Processes

The structure of a document space perspective, as described above, is designed to grow reflecting the actions performed on the current snap. Construction proceeds through users' implicit and explicit operation. This model defines operations for anchors, snaps, and threads.

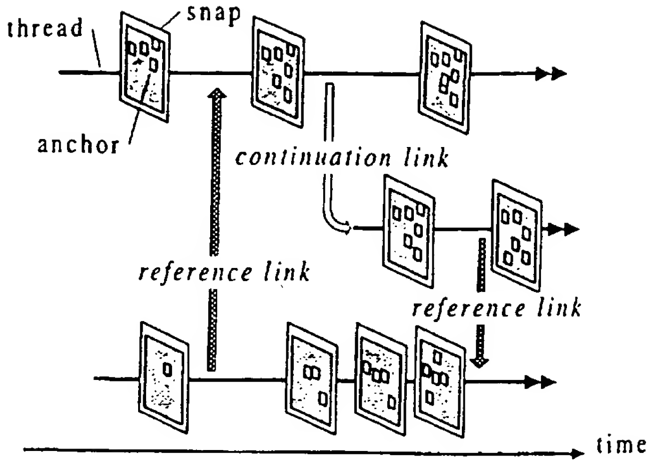

Figure 2: Temporally-threaded workspaces

Anchor operations enable users to manage anchors on the current snap and affect the state of the snap. They include operations for creating and removing an anchor, importing an anchor from another thread, and accessing a document via an anchor. Snap operations enable users to manage snaps in a thread and affect the structure of the thread. They include operations for taking (recording), retrieving, and synthesizing snaps. Thread operations enable users to manage thread and affect the structure of the document space perspective. They include operations for creating an (independent) thread, creating a successive thread, and referencing another thread. Any operation may be invoked explicitly by users; some of them are executed without direct invocation from users.

The unique feature of our model is the function of implicitly capturing an activity, and generating new state from the past states to proceed with the activity.

Implicit construction. The implicit construction mechanism is designed to get richly-structured activity-based perspectives without extra efforts from the worker. A workspace thread usually provides the latest snap in the thread to maintain coherence of the activity's workspace. The snap includes the set of anchors related to the current state of user's activity, and it reminds the user what he or she should do and helps to keep the activity on track. We expect users to create a workspace thread for performing an activity, and create, import, and remove anchors referencing documents on the latest snap as the natural actions for progress of the activity, with direct invocation of corresponding operations. Several operation sequences cause implicit invocation of other operations. For example, removing anchors from the current snap, switching the current snap to a past snap, or quitting the session for performing the activity invokes the operation for taking snap to record the state before executing these operations. Importing an anchor from a snap of another workspace thread invokes the operation for referencing the source thread.

Snap configuration. Like other version management systems, our framework has the capability to store states of workspaces as snaps and to recover the snaps when needed. However, the key feature of our framework is not recovering, but preparing new environments by using past snaps.

Since a snap is defined as a set of anchors, we can get a new snap by applying set-based functions, such as intersection and subtraction. The operation for synthesizing snaps applies a combination of set-based functions to past snaps according to user's instructions, and makes the result the current snap of a thread with which the user starts a new activity. Retrieving a past snap can also works to establish the environment for starting a new activity, if the user selects a past snap and makes it the current snap of a thread.

PRESERVING STATES OF EXTERNAL DOCUMENTS

In this section we introduce a mechanism for managing the contents of external documents referenced via anchors. Since the model we have introduced in the previous section handles only set of anchors, it records the internal context of activity. It does not account for changes to the contents of the referenced documents. If the document referenced by an anchor is changed, the context is not correctly recovered and is less helpful for understanding the past context. The solution for coping with this problem should satisfy following requirements:

- It must provide some control over external documents managed by people who are indifferent to workers' activities. Since the Internet provides distributed hypertext, the document may belong to a site run by an unrelated organization.
- The relationship to the internal context should be maintained. When accessing through past snap, the state of document at the time when the snap was taken should be retrieved.

The mechanism that we have introduced is a special kind of WWW proxy that relays the document access request from the client to the external or internal site that contains the requested document. The special feature of this proxy is that it records the contents of the returned documents. Figure 3 illustrates the mechanism of preserving and recovering the past state of external documents. When accessing a document via a locator, users can include an indication of whether they want the current version of the document, or the version of the document at a specific time. It may be designed to store the accessed documents regardless of whether they are external or internal. This characteristic would be important if users wish to manage internal and external documents in the same way.

When used in combination with the temporally-threaded workspace model, this framework helps users get a temporal-spatial perspective on the document space related to their activities. Although the proxy accumulates the accessed data, for technical reasons it is not easy for users to send the date information to get the necessary state of the document. Even if we can retrieve a list of all versions of the document, it is still difficult to select the correct version because the date is not a good description of the document version. A more effective clue for retrieving a specific version of a document is the state of the activity from which the document was accessed. Accessing a document via a snap sends a request including the snap's date to the proxy.

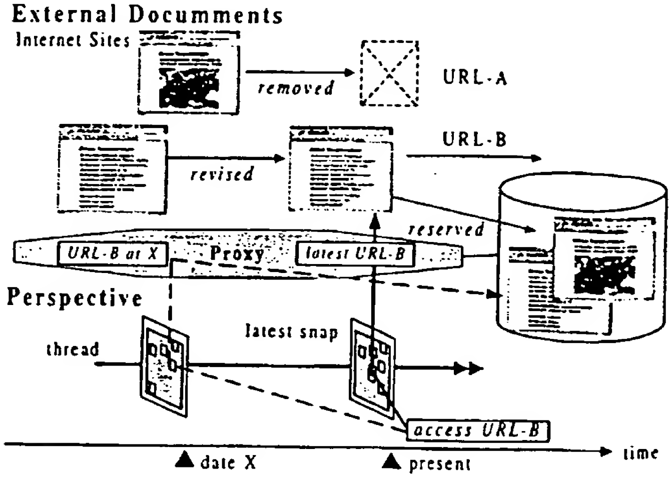

Figure 3: State preserving of external documents

EXAMPLE

Figure 4 shows an example of a spatial-temporal perspective including two interrelated workspace threads. Suppose that a company has decided to develop a new product that reduces total CO_2 emissions drastically. Dr. A is a chief researcher of a corporate lab in charge of designing the product. Ms. B is a planner from headquarters in charge of marketing the product. Workspace threads "Save the Earth" and "CO2 Reducer" capture document spaces related to each worker's activity.

The initial step in their activities is to understand the circumstances surrounding CO_2 emissions. Dr. A searches for literature to find current technology for reducing CO_2 emissions (1,2); gathers data on the total emissions from the company's latest products by simulation in his or her lab (3). Ms. B locates public sites that publish the latest information about regulations and the government's direction (4). They exchange significant information. In this example, Ms. B imported the anchor for referencing the external source from Dr. A's thread (5). The act of importing an anchor implicitly defines a reference link from thread "CO2 Reducer" to thread "Save the Earth". As successive tasks, they identify important information and exchange opinions to decide the specifications for the products. After the specifications have been established (6), Dr. A starts designing the products "Reducer Design" (7), while Ms. B starts to consult with the Sales department to get customer requirements "Reducer Requirement" (8).

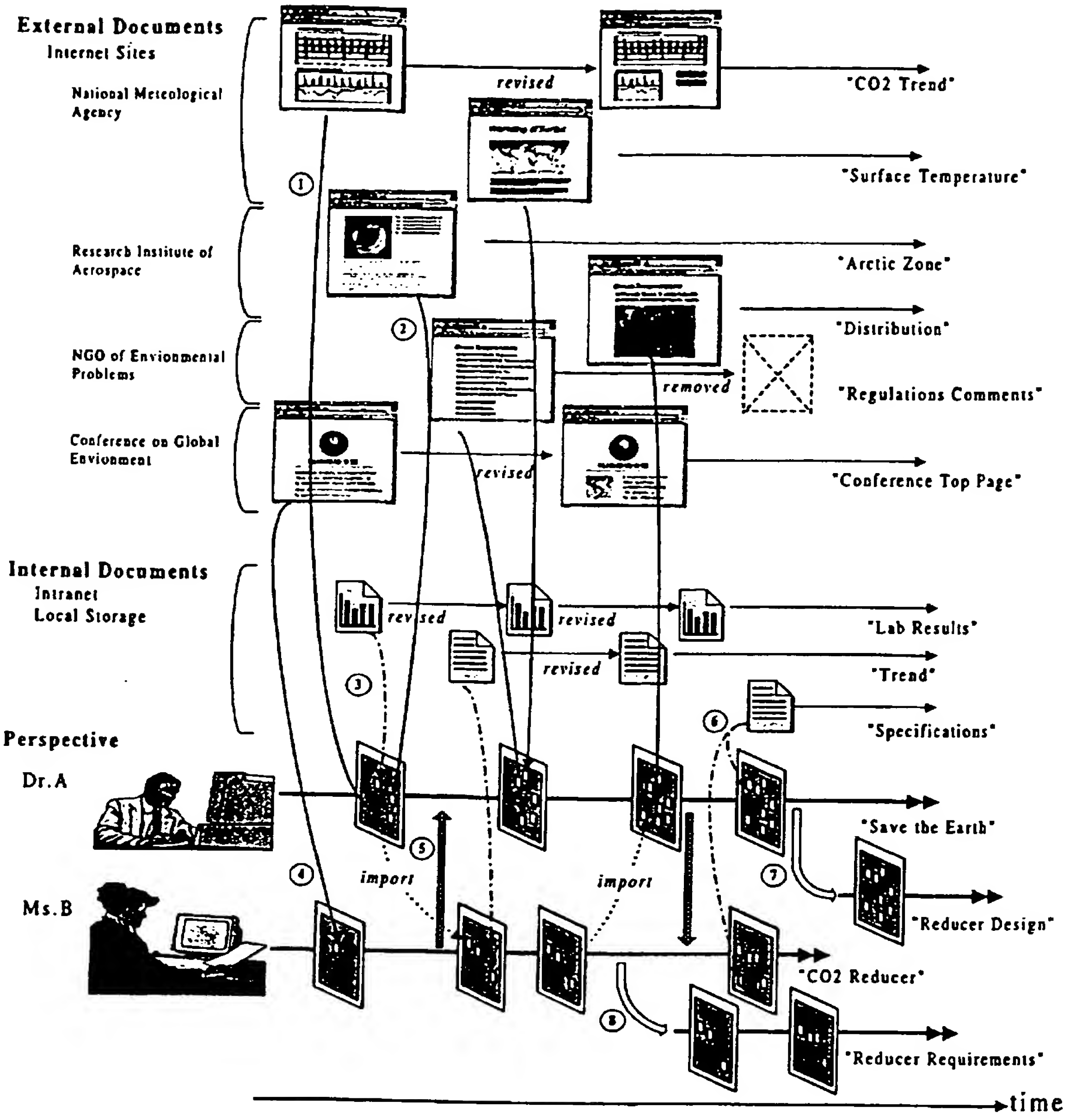

Figure 4: Example of activity-based perspective

The perspective on the document space formed while executing their activities helps users understand and retrieve necessary knowledge. In particular, it is advantageous for dealing with unpredictable situations. For example, when Dr. A or Ms. B must return to an earlier stage because of a change of regulations, or when another team must start designing another version of the products. From the activity-based perspective of the document space, we can get following knowledge:

- The circumstances in which the specifications were decided
- Proper sites to get trends of international CO_2 emissions control
- Current and past product data about CO_2 emissions
- Department contacts necessary for help with the project
- The necessary lead time for drafting new specifications
- The states of the successive activities such as product design

With conventional information management tools, maintaining such information would require users to exert extra effort to record the ever-changing work situation. They have to spend time to manage information that will rarely be used, because many of these situations are unpredictable. Otherwise, such activity-related knowledge would be lost as time passes.

IMPLEMENTATION: INTERLOCUS AND PACKRAT

We have implemented our activity-based perspective model of a document space as an extension of the WWW document space. We implemented *Interlocus*, a client/server system based on temporally-threaded workspace model, and *Packrat* [13], a WWW proxy server preserving external document states. Both systems are implemented with Java. Figure 5 shows the environment of our experimental systems.

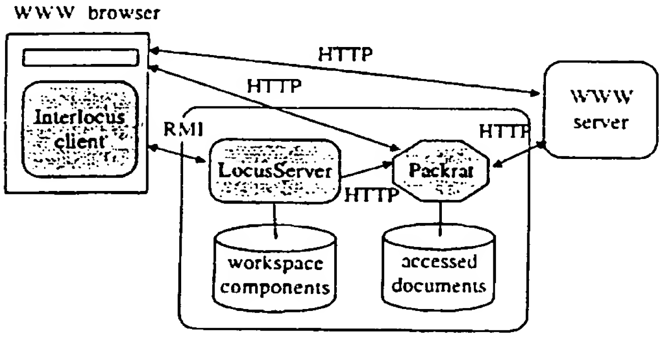

Figure 5: System environment

Interlocus

Interlocus consists of *Locus Server* and *Interlocus clients*. The Locus Server is the server that maintains objects and enables operations for realizing the temporally-threaded workspaces. Since Interlocus client is implemented as a Java applet, users can use Interlocus within most standard WWW browsers. The server and client exchange necessary Java objects via RMI. The interface of Interlocus client is designed to help users recognize the changes of focus in the given document spaces.

Spatial-temporal view. A novel feature of the Interlocus client is the user interface of the *activity view*, which presents a spatial-temporal view of a workspace thread. This tool consists of the *spatial frame* and the *time line frame*. The spatial frame displays anchors on a two-dimensional space. It describes the relations among a set of anchors. Figure 6 shows an example of an activity view. It shows the final snap of Dr. A's workspace thread "Save the Earth". A spatial frame holds two types of graphical elements, icon and region. An icon shows an image to reference documents or programs. A region shows a wider image that fills a region in which icons can be placed. Regions, as well as icons, are represented as anchors. Users can change size and image of each anchor at will. The set of these anchors represents the state of the document space, reflecting the stage of the activity. Figure 6 presents 8 icons and 3 regions that comprise a background under the icons. In the spatial frame, users can add anchors, remove anchors, and get documents via anchors. The time line frame presents the snaps comprising the workspace thread. By selecting one of the snap's symbols, the activity view displays the state corresponding to the selected snap. Figure 6 presents 6 snaps at Nov. 20 and 12 snaps at Nov.21, and indicates the latest one is the displayed snap.

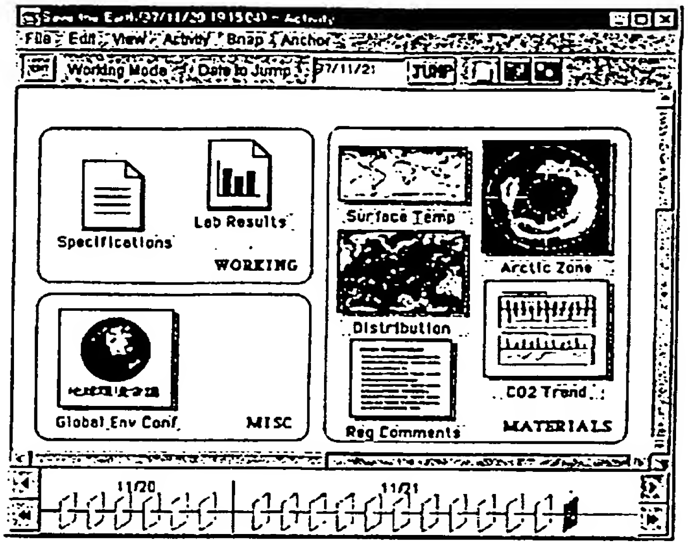

Figure 6: Activity view of an Interlocus client

Packrat

Packrat is a proxy server that records the state of the document contents as accessed and enables users retrieve the state when required. Packrat receives requests from clients via HTTP. If the client requests the document as it was at a specific past date, Packrat retrieves and returns the document data that was recorded before the specified date. If the request does not have date information, Packrat retrieves the latest version of the document located by the URL. Since it works as a WWW proxy server, users can use Packrat from a standard WWW browser by setting Packrat as the proxy server in the browser's preferences. Traversing the documents in the Internet document space with a standard WWW browser accumulates the accessed documents in Packrat's data storage.

Combination of Interlocus and Packrat. Interlocus and Packrat can work independently, but by using them in combination, workers can get a sound perspective of their activities. The Interlocus snaps describe the context of the activity and bound with the creation date. When the operation for getting a document is invoked, a request is sent to Packrat with the bound date of the snap.

Experiences

We have started daily use of Interlocus and Packrat in a small group. Since we have not yet steadily used the systems long enough to analyzing the logged data, we cannot conclude whether the perspective we introduced has much advantage over conventional hypertext.

However, we can present some findings from our experience using these systems.

- The combination of spatial frame and time line frame is useful to help users to understand the progress of activities. The background image, which comprises regions, and icons placed on the background intuitively shows the state of an activity. If the background image depicts the structure of an activity, the placement of icons on the background can represent a specific state of the activity. Further, the state (snap) reserving mechanism helps users maintain the current snap properly by reducing mental barrier to remove unnecessary anchors. Sequential access with time line frame makes the movement of anchors on the structure shown in the background natural to the human sense of motion, and therefore it helps users recognize the changes to the activity.
- Awareness of progress of related activities activates unexpected exchange of information and collaborative work. Each member can be aware of progress of others' activities with activity views. Awareness of progress in searching for information has affected the searching activities of others in several ways, for example to start searching at the site that the other member found.

DISCUSSION

Workspace thread as a unit for representing activity

We have introduced the temporally-threaded workspace model to capture the information space related to a given activity. The current version maintains the set of documents, which must be identified by URLs. It is controversial whether this set represents the state of an activity and can be used for reminding, recognizing, or reusing information to progress in corresponding activities.

Some users pointed that we need to capture telephone conversations or in-person discussions. Our model does not exclude these types of data, since multimedia data can be a document, in the wide sense of document, and the environment to manage such data will be soon available with WWW technology. From the viewpoint of cognition, it is more important whether a unit captured as a workspace thread is the proper unit as an activity. The current version of our model is designed to extract the main activity when making documents like reports, papers, and charts are central to the tasks. However, when a presentation or a lecture is central, our framework captures only activities of the preparation and review of these events. For capturing such dynamic events as activities, we might need to provide a different metaphor than desktop like workspace. Users do not think that they are working in a desktop while making presentation.

Cost of management of the past states

To maintain the dynamic perspective of the document space, both Interlocus and Packrat manage the past states of activities. The cost of maintaining past data should be discussed. Each system has different characteristics. Locus Server accumulates snaps at the events of changing a set of anchors and switching current snap. Since the body of snap is a set of URLs, the cost of recording is not expensive. In contrast, Packrat accumulates the document content data, and thus it consumes far more storage.

Although it might seem that there is a limitless amount of WWW data, we can expect the amount of data has a realistic limitation because of human physical and cognitive capability for handling documents. The short-term experiments of 54 days, Packrat has received 14,449 accesses, recorded 11,923 URLs and consumed 105 MB[13]. We are optimistically thinking that the progress of storage media will solve this problem. So far as the limitation works, we can even enlarge the range of recording (e.g. include directly linked pages from accessed pages). However, it will depend on the future widespread popularization of multimedia data. If the increase of multimedia data usage is faster than the progress of storage technology, we would need to use data compression.

Comparison to related work

We again discuss the related work from technical viewpoints. The temporally-threaded workspace model is an extension of the concept of Rooms. We follow the concept of preparing a two-dimensional space that maintains tools and documents for each activity. Rooms provides a room that has common tools and documents for each of the categories of activities. In the sense of providing an environment in which each activity proceeds, our approach is also similar to the concept of the activity spaces of Sepia and COWFISH. The activity space is defined as the space to support a category of tasks such as getting information, augmentation, and presenting documents. Both rooms and activity spaces require users to specify categories of their tasks and prepare necessary functions to support each category. These approaches work well when users' main tasks are recognized as combination of predefined activities. However, it is not always easy to maintain the common functions and documents, because each specific instance of task often has special requirements. Further, many users lack the motivation to maintain the common functions for future use when they have other urgent tasks.

On the other hand, in the temporally-threaded workspace we introduce, the environment is related to each specific instance of a task, such as preparing a report about international CO_2 emissions control, discussing the design of environment safer products, and planning customer support using Internet. Workspaces record the progress of specific instances of tasks, and provide functions for exploring and finding necessary documents and tools from the accumulated records of activities. Our approach is advantageous for our intention to capture a wider range of workers' activities including ad-hoc tasks.

As for the user interface, workspace has features that combine the desktop metaphor and spatial hypertext. Like

VIKI, we use spatial placement of images as clues for representing perspectives of documents. However, we did not introduce features using structural components. Instead, we adopt a desktop feature that enables users to place icons referencing to a document on two-dimensional space. In addition to this feature, users can define and place regional anchors that provide a richer representation of their perspective on the set of documents. Also the mechanisms for recording snapshots and structuring threads help users to keep the current snap of each workspace concise. The simple desktop metaphor is desirable to minimize the barrier for users to start using our system, since it is essential for our framework to capture as many workers' activities as possible. Even so, the structural components VIKI introduced would be useful to extend our framework. They would enhance the ability to express the user's perspective on complicated information spaces.

The concepts and mechanisms that we introduced for managing changes to workspaces and external documents are based on a general model of version and configuration management of hypertext discussed by Østerbye and Haake. Our approach is distinguished from these version management systems by the purpose and range for maintaining states. Our purpose of the management of state is not to recover the state of the hypertext structure, but to provide perspective of information space that varies reflecting internal or external changes, and further, to support users in proceeding with their tasks by preparing new snap state. We have extended the range of management from local hypertext to distributed hypertext by providing WWW proxy mechanisms for preserving states of external documents.

FUTURE WORK

Supporting collaborative work

The framework we have introduced is a model of activity-based perspective on document space as an extended concept of hypertext. The essential issue of our framework is to provide functions that exploit the introduced structure of threaded workspaces with realistic usability. Through our short-term experiments we found that the structure is useful to control exchange of information among related activities. To support such collaborative advantages, we further have to deal with the following issues:

How to understand others' activities. To help users comprehend the spatial-temporal perspective on document space, we provide a user interface combining a spatial frame and a time line frame. It works well for understanding a document space related to the activity of worker's own or tightly collaborating colleagues. However, it is more challenging goal to get perspective of activities of others who are distant from the user in spatial, temporal, and organizational terms.

How to find and track others' activities. To work collaboratively with other members, we need a way to know where and how a related activity is proceeding. We think the links connecting workspaces threads can guide the selection of associated activities and the set based operations among snaps can be used to represent their collective progress.

How to get reusable information from the past. To use the accumulated records of activities effectively, we have to provide a way to find the reusable part of information. We think that somewhat new ways to create indexes when important events occur and a way to specify event criteria are useful.

How to choose part to publish. If many users share workspace threads, the owner may wish to limit the region to be published to other workers. Since the system enables users access past data, the access rights as they were in the past should be applied. Users might decide what snaps should be accessible and how they should be presented when they publish their workspace threads.

Seamless integration in the WWW

The current implementation adopts original data structures that are used to provide activity-based perspectives on the WWW document space. Further, both Packrat and Interlocus use extended interfaces for exchanging data. To extend the space of activity to other sites, we need a way to implement the systems using standard WWW technology. The conceptual model we have described is not limited to the implementations we have discussed. All of the components and links can be described by using standard HTML documents. However, for interactive operations like editing snaps, the HTML browser must be extended. Further, for managing a series of snaps in a workspace thread, storage requires the functionality of HTML version management.

For Interlocus we defined special components and communication between server and clients using objects. Current HTML documents are not flexible enough to represent workspace components. The simple solution is to provide a way to read (but not write) other sites. To Locus Server we can add functionality to generate HTML presentations of the workspaces. The long term solution is implementation with XML format. Since XML has richer capability to describe various links, the necessary extension for our framework will be possible.

CONCLUSIONS

Today the WWW-based hypertext is rapidly growing, soaking up all kinds of information generated by human activities. The problem of cognitive overhead for understanding hypertext, which is one of the classical problems when hypertext research started, still remains in a different form. The question of "what changes have happened to a hypertext space" has become "what is going on in the world". Even though the WWW document space holds a great deal of information about activities in the world, there are few clues to associate documents with the activities that are generating them.

In this paper we have presented a framework for providing activity-based perspectives on document spaces. The novel features of this framework are providing knowledge workers' workspaces that clip subspaces related to their activities, and adding the time dimensions. So far we have only short-term experience with our initial experimental system that consists of Interlocus and Packrat, which covers only limited varieties of intellectual activities at present. However, we have experienced new perspectives on document spaces related to activities with a sense like rising a 3-dimensional figure from a stereogram. For living in future WWW-centric world, it would be helpful to obtain other senses for recognizing the WWW document spaces. The activity-based perspective and its spatial-temporal views presented in this paper could be a starting point for exploring these senses.

REFERENCE

1. Bentley, R., Horstmann, T., Sikkel, K., and Trevor, J. Supporting Collaborative Information Sharing with the WorldWide Web: The BSCW Shared Workspace System. In Proceedings of WWW4 (1995).

2. Bentley, R., Appelt, W., Busbach. U., Hinrichs, E., Kerr, D., Sikkel, S., Trevor, J. and Woetzel, G., Basic Support for Cooperative Work on the World Wide Web, in International Journal of Human-Computer Studies, 46, 6 (June 1997).

3. Campbell, B. and Goodman, J.M. HAM: A General Purpose Hypertext Abstract Machine, Commun ACM, 31,7 (1988) 856-861.

4. Card, S.K. et al. A Multiple, Virtual-WorkspaceInterface to Support User Task Switching. In Proceedings of CHI+GI (1987).

5. Cellary, W., Vossen, G., and Jomier, G. Multiversion Object Constellations: A New Approach to Support a Designer's Database Work. Engineering with Computers, 10 (1994) 230-244.

6. Conklin, J. Hypertext: An introduction and survey. IEE Comput. 20,9 (Sept. 1987), 17-40.

7. Haake, A. CoVer: A Contextural Version SErver for Hypertext Applications. In Proceedings. of ECHT'92, (Nov. 1992) 43-52.

8. Haake, A. and Haake, J.M. Take CoVer: Exploiting Version Support in Cooperative Systems. In Proceedings of INTERCHI'93 (Apr. 1993) 406-413.

9. Haake, A., Under CoVer: The Implementation of a Contextual Version Server for Hypertext Applications. In Proceedings of ECHT'94 (Sept. 1994) 81-93.

10. Haake, A., and Hicks, D. VerSE: Towerds Hypertext Versioning Styles. In Proceedings of Hypertext'96 (Mar. 1996) 224-234.

11. Halasz, F. Reflections on Notecards: Seven Issues for the Next Generation of Hypermedia Systems. Commun ACM, 31, 7 (1988) 836-852.

12. Halfhill, T.K., Good-Bye, GUI Hello, NUI, Byte, 22, 7 (1997) 60-72.

13. Hashimoto, S., Gudmundson, T., Hazama, T., Takeoka, M., Nomura, T., and Hayashi, K., Packrat: A Proxy Server that allows retrieval of previously-browsed web pages. In Proceedings of Japan World Wide Web Conference '97 (Dec. 1997).

14. Hayashi, K., and Sekijima, A. Mediating Interface between Hypertext and Structured Documents. Electronic Publishing 6,4 (1994) 423-434.

15. Henderson, D.A., Jr. and Card, S.K. Rooms: The use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. ACM Trans. Graphics 5,3 (July 1986), 211-243.

16. Katz, R.H. Towards a Unified Framework for Version Modelling in Engineering Databases. ACM Computing Surveys, 22,.4 (1990) 375-408.

17. Malone, T.W. How do people organize their desks? Implications for the design of office information systems. ACM Trans. Info. Syst. 1,1 (1983), 99-112.

18. Marshall, C.C., and Shipman, F.M. III. Searching for the Missing Link: Discovering Implicit Structure in Spatial Hypertext. In Proceedings of Hypertext'93 (Nov. 1993), 217-230.

19. Marshall, C.C., and Shipman, F.M. III. Spatial Hypertext: Designing for Change. Commun. ACM 38,8 (Aug. 1995) 88-97.

20. Marshall, C.C., and Shipman, F.M.III. Spatial Hypertext and Practice of Information Triage. In Proceedings of Hypertext'97 (Apr. 1997), 124-133.

21. Streitz, N., Hannemann, J., Thüring, M., From Ideas and Arguments to Hyperdocuments: Travelling through Activity Spaces. In Proceedings of Hypertext'89 (Nov. 1989) 343-364.

22. Streitz, N., Haake, J., Hannemann, J., Lemke, A., Schuler, W., Schutt, H., and Thuring, M. SEPIA: A Cooperative Hypermedia Authoring Environment. In Proceedings of ECHT'92 (Nov. 1992) 11-22.

23. Østerbye, K. Structural and Cognitive Problems in Providing Version Control for Hypertext. In Proceedings of ECHT'92 (Nov. 1992) 33-42.

24. Wang, W. and Haake, J. Supporting User-defined Activity Spaces. In Proceedings of Hypertext'97 (Apr. 1997), 112-123.